Robotica (2011) volume 29, pp. 555–562. © Cambridge University Press 2010
doi:10.1017/S0263574710000457

Fault detection and isolation in cooperative mobile robots using multilayer architecture and dynamic observers

R. A. Carrasco†,*, F. Núñez‡ and A. Cipriano‡

†*Department of Industrial Engineering and Operations Research, Columbia University, New York, USA*[1]
‡*Department of Electrical Engineering, Pontificia Universidad Católica de Chile, PO Box 306, Santiago 6904411, Chile*

(Received in Final Form: July 16, 2010. First published online: August 12, 2010)

SUMMARY

Mobile robot systems are being used more often in tasks that protect human operators from dangerous environments, but these benefits can be easily lost if the robots spend much of their time being repaired. This implies that any increment in their reliability will also improve their benefits. One way to achieve this is by adding redundant elements to the robot, but this adds complexity and cost to the design. On the other hand, cooperative mobile robots formed by members with the same basic structure provide a natural redundancy of elements, which may be used for reliability improvement. This work presents an architecture that takes advantage of the analytical and sensor redundancy present in groups of cooperative mobile robots in order to increase the reliability of the whole system. First, the design of the architecture is portrayed and the faults to be detected are described. The different layers of the system are then explained and analyzed using several simulations to test their capabilities and limitations. Finally, the experimental results on a group of small mobile robots are shown, validating the results delivered by simulations. These results show that the proposed architecture is able to detect and isolate correctly most of the faults tested.

KEYWORDS: Cooperative systems; Distributed detection; Fault detection and isolation; Mobile robots.

1. Introduction

The use of robotic systems in environments hostile for human beings or in dangerous tasks, such as land mines extraction, and rescue operations, has increased over the last years, making the work safer to human operators. However, the advantages of operating with robotic systems are cut back when faults occur as they disable the robot in some of its functions, or in worst cases, they make the robot unable to work at all. Recent studies show that the mean time between failures is less than 20 h for field robots,[1] after which they require repairs, consuming time and resources. This makes an increment in reliability a necessity in order to improve mean time between failures and also to reduce the mean repairing time.

Two main methods have been used to increase the reliability of robots. One method is to add mechanical and/or sensory redundancy, which regretfully increases the construction costs significantly. The other method for increasing reliability is to add fault detection and isolation (FDI) systems. These systems identify present problems and thus reduce time and effort needed for repairs,[2,3] or help in the incorporation of fault tolerant controls, which modify control objectives to accommodate present faults and continue with normal operation if possible. This work focuses in the latter; the aim is to develop a FDI architecture for cooperative mobile robots, capable of providing information that can then be used either to achieve a fault tolerant system or to reduce the time needed to restore them to normal functionality.

Developments on fault analysis appeared in the early 1970s with fault detection, isolation, and identification systems. Fault detection only indicates that a fault is present on the system; fault isolation follows the fault detection process and determines the exact location of the fault, whereas fault identification (also called fault diagnosis[2]) can determine the location and size of the fault.[4,5] Nowadays, developments on FDI and fault detection and diagnosis (FDD) cover a wide range of applications that go from the water tank benchmark system, to more complex processes such as airplanes, automotive systems, and robots.[6–9]

There are a wide range of techniques to achieve FDI, which most of the time are tailor made for each specific application. These techniques include the use of tools such as fuzzy logic, neural networks, wavelets, and Kalman filters, which are used to analyze and detect malfunctions.[10–12]

One of the most effective approaches to achieve FDI on mobile robots is based on multiple models and a bank of observers, where each model has one of the possible faults embedded and the corresponding observer estimates a state vector for the system according to that.[3] As Kalman filters give an optimal estimation of these state vectors if the noise is normally distributed,[13] it is an excellent tool for this approach. In this technique, a bank of Kalman filters is implemented, exploiting the analytical redundancy present. Each Kalman filter estimates the state vector, using as model for the system one of the possible operating modes (normal and faulty modes); by comparing the estimation against the measurements vector, a residual vector is formed for each Kalman filter, which are then analyzed to determine which fault has occurred. This technique is described in detail in the work of Maybeck and Hanlon,[10] where it is applied through

* Corresponding author. E-mail: rax@ing.puc.cl
[1] Work done while he was at Pontificia Universidad Católica de Chile.

simulations to aircraft FDI, using multiple hypotheses to isolate each fault.

This fault detection algorithm, applied to mobile robots, is presented in Roumeliotis *et al.*,[14] where only two possible faults are analyzed: a reduction in the radius of one tire and a periodic bump, without processing the information in the residuals extensively. In Roumeliotis *et al.*,[15] the work in Roumeliotis *et al.*[14] is extended through the use of multiple hypotheses to isolate sensor faults. The probability of each hypothesis is calculated using the residuals of the corresponding Kalman filter, showing good results in FDI. Regretfully, in this work there is no description of the isolation criteria used to determine which fault has occurred. A similar bank of Kalman filters is used in Goel *et al.*[16] to determine faults (on sensors and actuators) on a four-wheel robot, but in this case the residuals are processed through a neural network to isolate the faults. Another isolation method is presented in Washington,[17] where the bank of Kalman filters is combined with a Markov model representation to identify the faults through probability calculations. Although these works present important advances in FDI, they detect faults only in speed related sensors (e.g., gyros and encoders), leaving behind other important sensors such as sonars, GPS, and magnetic compasses, where FDI is a more complex task.

Other approaches to FDI in mobile robots include the use of more advanced filtering techniques to identify the faults, incorporating nonlinear robot dynamics, non-Gaussian noise, geometrical constraints, and statistical analysis.[18–20]

Also, aiming toward an improvement in the reliability of robotic systems, several researchers have proposed a multiple robot approach.[21–26] In Michaelson and Jiang,[21] the authors explain how the redundancy present in cooperative mobile robots can be used to increase the robustness of the group, thus improving the efficiency, but no fault detection system is described, vital thing to the performance of the method. In the same line of study the ALLIANCE architecture presented by Parker[22] shows a simple fault detection system for cooperative robots based on behavioral programming, but fault detection process is limited to detect when a robot has suffered a fatal failure and the authors indicate that it presents a slow response. Using another approach, the work developed by Tinós *et al.*[23] shows a distributed localization scheme for resource limited mobile robots. The algorithm takes advantage of measurement redundancy to improve the localization of each robot and, at the same time, it is used to detect which localization measurements are incorrect, eliminating them from the system. The work in Tinós *et al.*[24] describes a simple FDI method for cooperative manipulators, which also uses measurement redundancy. Their experiments show that faults are correctly detected over 90% of the time, whereas in 68% of the experiments faults are properly isolated. Heredia *et al.*[25,26] present a cooperative approach for differential GPS fault detection in unmanned aerial vehicles (UAV). The proposed scheme uses artificial vision-based relative position measurements, aiming to detect wrong GPS absolute position measurements. The approach is based on the fact that in multi-UAV missions, it is possible to take advantage of the capabilities that the team of UAVs offers, to augment each of the individual FDI systems. Different UAVs may identify, using their cameras, common objects in the scene providing an alternative relative position measurement. Results show that the cooperative scheme is able to detect differential GPS faults that are indistinguishable using only local FDI systems. Regretfully, with exception of Tinós *et al.*,[24] none of the works above mentioned contain an analysis of fault detection limitations, nor of the efficiency of the isolation (number of false positives or wrongly isolated faults), so no reliability comparisons can be made among the different methods.

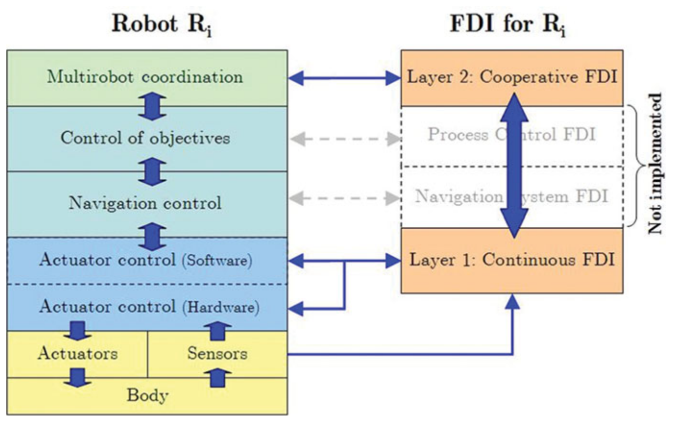

Fig. 1. Control and fault detection and isolation structures for robot R_i.

With the same objective, multiple layer approaches have also been used to achieve FDI in different classes of robotic systems (systems with different levels of resources), so the FDI system can be adapted depending on the redundancy that exists,[27,28] but the idea of having a cooperative layer within the architecture has not been implemented yet.

This work presents a layered architecture for FDI on cooperative robots that combines the advantages of single and multiple robots fault detection mechanisms, where the different layers can be implemented depending on capabilities and resources of the robots. The proposed architecture combines existing methods for single robot FDI, where local information is used in order to detect the presence of faults, with the ideas present in cooperative robot FDI systems, where additional information obtained from multiple robots is used for detecting faults in any of the group members; yielding an architecture capable of detecting a wider range of faults in comparison with local information-based FDI systems. The idea behind the multiple layer approach is to take advantage of the different levels of information, control, and redundancy that exists within the control structure (Fig. 1), tailoring each layer in the FDI system according to the level of information available at its corresponding level in the control structure and thus allowing for an efficient use of the information. However, it is probably impossible to create a single-general FDI system that can be successful on any group of mobile robots, as different robot systems have different capabilities, sensors, actuators, etc.

Figure 1 illustrates the division made on the control structure for each robot R$_i$, according to the level of control and information, and also shows the interaction with the layers of the FDI system. The control structure is divided into 5 main layers. First, the physical layer contains the body, sensors, and actuators needed. The Actuator Control Layer

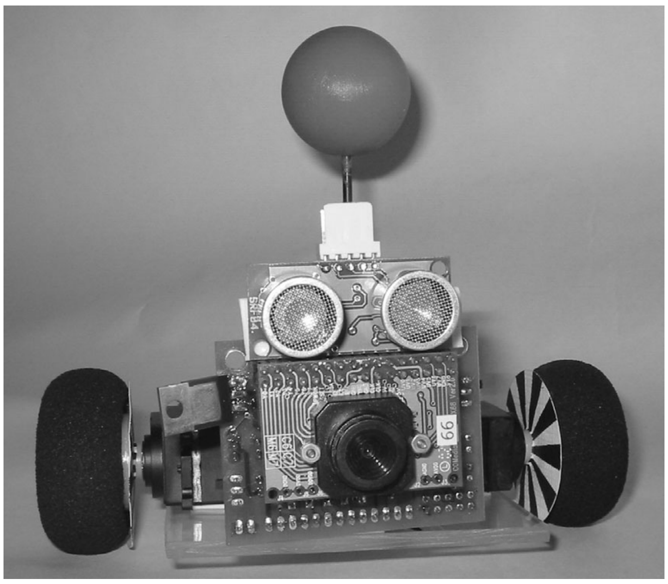
Fig. 2. Mobile robot used with marker.

is the interface between the hardware and the navigation, controlling the speed of the wheels in order to follow a determined trajectory. Next, the Navigation Control is dedicated to design the trajectories needed to achieve the different objectives. The highest control layer for a single robot is the Control of Objectives Layer, which designates the tasks that must be done and where the robot must go in order to do them. Finally, in cooperative robots the Multirobot Coordination Layer is added, which can be either centralized or distributed among the robots, and is the one that designates the objectives for each robot to achieve a common goal.

This work is divided into five sections. First, Section 2 presents a description of the robot system over which the FDI architecture is implemented describing the faults each layer can detect and isolate. Next, Section 3 describes the first layer of the architecture, presenting the method used and an analysis of the fault detection capabilities and limitations. Section 4 continues with the description of the second layer, indicating how the cooperative robots approach is used. Finally, Section 5 shows experimental results for each layer.

2. System Description

2.1. Robot simulation

The simulations were done in Matlab, using a mathematical model, which includes the kinematic and dynamic equations of each robot. The sensor readings were also simulated in the model by adding Gaussian noise to the measurements. The standard deviation of the noise added was equal to 10% of the maximum value of the measurement, which was above the range of what was observed in the real sensors used in the implementation.

2.2. Cooperative robot system

A group of three small homogeneous mobile robots, as the one shown in Fig. 2, built at our university, were used to test each layer of the architecture and validate the data obtained through the simulations. Each robot moves using two independent actuated wheels, allowing differential steering, and are controlled by two low cost 8-bit microcontrollers. Each robot is equipped with optical encoders on both wheels to achieve relative localization and a digital compass to measure the heading angle. For navigation purposes, each robot has a frontal sonar and a low resolution CMOS camera. For this work, the CMOS camera is only used to recognize other robots, which was done by identifying a red ball they have on top.

2.3. Faults

As different faults implemented need different levels of information to be isolated, faults are divided into two groups: those that can be continuously monitored on a single robot and those that can be detected through cooperation between them. Although some faults can be detected through both methods, they are assigned to the layer where it is easier to detect them.

For the first layer of the FDI architecture, seven different faults were considered: 1–2: slippage of one of the wheels, 3–4: one of the wheels gets stuck, 5: both wheels get stuck, and 6–7: one of the encoders fail (i.e., the velocity of that wheel is read as zero).

On the other hand, the cooperative layer isolates faults on sensors that are redundant in the robot team. This layer was designed to detect four different faults: 1: additive fault on the sonar, 2: the sonar gives a constant value, 3: additive fault on the compass, and 4: the compass gives a constant value.

3. Continuous FDI Layer

3.1. Method description

The use of multiple models has shown to be a good tool for continuous monitoring of faults in mobile robots. As all the faults this layer must detect can be modeled within a Kalman filter, a bank of eight Kalman filters is used: one for modeling normal operation (M_0), and seven for modeling the faults ($M_1 - M_7$). The basic structure of each model M_i is as follows:

$$M_i \begin{cases} X_{k+1,i} = A_i X_{k,i} + B_i U_k + w_{k,i} \\ Z_{k+1,i} = C_i X_{k+1,i} + v_{k,i} \end{cases}, \quad i = 0.7. \qquad (1)$$

In (1), $X_{k,i}$ is the state vector for the robot at time k using model i, whereas $Z_{k,i}$ is the measurement vector of that model at time k. The matrices A_i, B_i, and C_i are the state equation matrices for model i, and U_k is the control input at time k. The process and measurement additive white noise at time k are represented by $w_{k,i}$ and $v_{k,i}$, respectively.

To obtain an optimal estimation of the state and measurement vectors of each model M_i, a Kalman filter[13] is used. First, the state and measurement vectors are estimated, assuming that no noise is present., and the covariance matrix $P_{k+1,i}$ is estimated using the noise covariance matrix Q_i:

$$\left.\begin{aligned} X^-_{k+1,i} &= A_i X_{k,i} + B_i U_k, \\ Z^-_{k+1,i} &= C_i X^-_{k+1,i}, \\ P^-_{k+1,i} &= A_i P_{k,i} A_i^T + Q_i. \end{aligned}\right\} \qquad (2)$$

Next, the estimations are updated using the new available measurement Y_{k+1}, and the Kalman gain $K_{k+1,i}$

$$\left.\begin{aligned} X_{k+1,i} &= X^-_{k+1,i} + K_{k+1,i}(Y_{k+1} - Z^-_{k+1,i}),\\ Z_{k+1,i} &= C_i X_{k+1,i},\\ P_{k+1,i} &= P^-_{k+1,i} - K_{k+1,i} C_i P^-_{k+1,i}. \end{aligned}\right\} \tag{3}$$

The Kalman gain is calculated as follows, where $S_{k+1,i}$ is the residual covariance matrix at time $k+1$ for model M_i, and R_i is the covariance matrix for the measurement noise in that model.

$$K_{k+1,i} = P^-_{k+1,i} C_i^T S^{-1}_{k+1,i}, \tag{4}$$

$$S_{k+1,i} = C_i P^-_{k+1,i} C_i^T + R_i. \tag{5}$$

Using these estimations, faults are detected by calculating the probability of hypothesis H_i to be true, which states that model M_i represents the actual operation mode of the robot. The conditional probability that hypothesis H_i is true at time $k+1$ is given by the following expression:[10]

$$P_{k+1}(H_i) = \frac{f\left(Y_{k+1}/\mathrm{M}_i, \left[Y_0^T..Y_k^T\right]\right) P_k(H_i)}{\sum_{j=0}^{7} f\left(Y_{k+1}/M_j, \left[Y_0^T..Y_k^T\right]\right) P_k(H_j)}. \tag{6}$$

In (6), $f(\cdot)$ is the conditional probability density function of the measurement Y_{k+1}, conditioned on the model M_i and the previous measurements. This is determined by the following:

$$f\left(\frac{Y_{k+1}}{M_i}, \left[Y_0^T..Y_k^T\right]\right) = \beta_{k+1,i} e^{-\frac{1}{2}D_{k+1,i}}. \tag{7}$$

With $D_{k+1,i}$ (the Mahalanobis distance at time $k+1$) and $\beta_{k+1,i}$ defined by following:

$$D_{k+1,i} = r^T_{k+1,i} S^{-1}_{k+1,i} r_{k+1,i}, \tag{8}$$

$$\beta_{k+1,i} = ((2\pi)^m |S_{k+1,i}|)^{-\frac{1}{2}}. \tag{9}$$

The parameter m used in (9) is equal to the number of elements in the measurement vector as the noise is distributed as a multivariate normal distribution. The residual $r_{k+1,i}$ is obtained as the difference between the measurements vector and the estimation of the measurement vector given by model M_i.

$$r_{k+1,i} = Y_{k+1} - Z_{k+1,i}. \tag{10}$$

It is important to notice in (6) that if the probability of a certain hypothesis reaches 0, it cannot return to another value. This problem is solved by artificially setting the minimum probability for any hypothesis to 0.0001.

To reduce the computational requirements of the Kalman filters, only three variables are taken into account for the state and measurement vectors: the rotation speed of the robot and the speed of each wheel, and hence the state vector is given as follows:

$$Y_k = [\dot{\varphi}_k \quad \omega_{1,k} \quad \omega_{2,k}]^T. \tag{11}$$

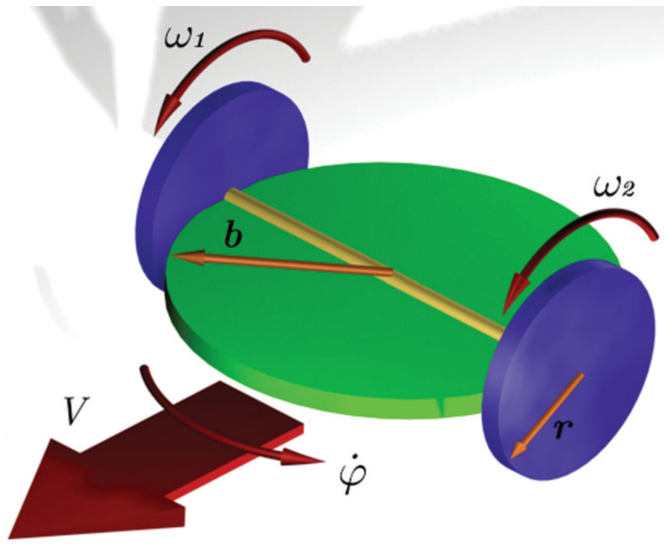

Fig. 3. Simplified scheme of the mobile robot used.

Figure 3 presents a simplified scheme of the mobile robot, where the main variables are shown. Each Kalman filter uses a simple kinematic relation, which is modified according to the operation mode.

$$\dot{\varphi}_k = \frac{\lambda r_1 \omega_{1,k} - \mu r_2 \omega_{2,k}}{2b}. \tag{12}$$

Equation (12) relates the speed of each wheel with the rotational speed of the robot. In (12), r_1 and r_2 are the radii of the right and left wheels, respectively, and $2b$ is the axle length. λ and μ are parameters used to represent the different faults. For M_0, $\lambda = \mu = 1$; for M_1, $\lambda = 0.4$ $\mu = 1$; for M_2, $\lambda = 1$ $\mu = 0.4$; for M_3, $\lambda = 0$ $\mu = 1$; for M_4, $\lambda = 1$ $\mu = 0$; for M_5 $\lambda = \mu = 0$; and for M_{6-7}, $\lambda = \mu = 1$ as the fault affects only the measurements and not the process. The value 0.4 used in the model for faults 1 and 2 was determined empirically, aiming to eliminate false alarms due to the small slippage that mobile robots always have.

3.2. *Fault detection and isolation*

Once the probability of each hypothesis is calculated, FDI is done by using thresholds. First, a fault is detected when the probability of H_0 is smaller than the threshold P_{DT}. This value is a free parameter that allows tuning, affecting the response time of the detection. Higher values of P_{DT} reduce the detection time, but the number of false alarms increases, whereas if the value is low, the detection takes longer but false alarms are reduced. Due to the effect of noise, the probability of H_0 can sometimes be lower than P_{DT} for some time intervals, creating a false alarm. To reduce this, detection is activated if the probability of H_0 is smaller than $P_{DT} = 0.01$ for three consecutive time intervals, thus reducing false alarms without significantly increasing detection time.

After fault detection is done, isolation is achieved by detecting which probability surpasses the threshold P_{IS}. If the probability of H_i is above $P_{IS} = 0.99$, it is assumed that fault i occurred. These values were determined through simulations, reducing wrongly isolated faults.

Table I. Average detection and isolation times for layer 1.

Fault	1	2	3	4	5	6	7
Detection [s]	1.1	1.3	0.7	0.6	0.7	0.3	0.3
Isolation [s]	1.1	1.3	0.8	0.8	0.7	0.3	0.3

3.3. Simulation results

The continuous FDI layer is tested through several simulations. The system is simulated 1000 times with random chosen operation modes (normal and faulty ones), setting on each simulation the sampling time at 0.1 [s]. This allows a statistical analysis of the layer performance.

Four different criteria are used to measure the performance: amount of false alarms, confusion matrix, and FDI times.

False alarms indicate the number of times that faults are detected while on normal operation. The simulations show that no false alarms appear because of the detection criteria.

The confusion matrix shows the relation between the faults that appear on the robot and the faults isolated by the FDI layer. The matrix contains as elements c_{ij}, the percentage of times when operating mode M_i is isolated due to hypothesis H_j, helping to determine the percentage of wrongly isolated faults. The following confusion matrix was constructed based on the results of the simulations:

$$Cf_{\text{Layer}_1} = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 98.1 & 0 & 0 & 1.9 & 0 \\ 0 & 0 & 0 & 0 & 97.6 & 0 & 0 & 2.4 \\ 0 & 0 & 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 100 \end{bmatrix}. \tag{13}$$

The results show that only a 0.4% of the total simulated operating modes are wrongly identified. This confusion only affects faults 3–4 (stuck wheels), which are isolated as faults 6–7 (encoder faults), respectively, as the effect of both faults is similar during the first time intervals. This confusion can be eliminated if the threshold used for isolation is raised, but that also implies a slower response.

Finally, Table I shows the average detection and isolation times. Due to the detection and isolation criteria, isolation takes slightly longer than detection in some cases. It can be observed that both detection and isolation take only a small number on time intervals.

3.4. Interaction with the control structure

The interaction between the FDI architecture and the control structure cannot only be used to inform that a fault is present. It can also be used to increase isolation capabilities, obtaining more information about the present state of the robot.

When the robot collides with an obstacle, the wheels can either slip or get stuck depending on the friction at that moment. In both cases, the FDI system will give a false alarm, as no real fault is present. If no further actions are taken, the robot can be considered disabled although it can still work.

To solve this problem a control routine is added, which helps to determinate if it is a real fault or a collision. Whenever a slippage or stuck condition is detected, instead of activating the fault detection flag, the FDI system asks for a change in the rotation direction of the affected wheel to the control structure. If the probability of H_0 returns to be high again, it means that the robot had collided with an undetected object, but if the probability of the faulty condition continues to be high, then a real fault is present. This simple algorithm creates a “virtual bumper sensor” that improves the fault isolation capabilities of the layer.

3.5. Method limitations

The biggest limitation of this method is that it must be possible to model the effects of the faults within a Kalman filter. If this is not possible, other isolation method must be used, as there is no residual covariance matrix available for the probability calculations.

4. FDI on Cooperatives Robots

4.1. Redundant sensor fault detection and isolation

Fault detection on redundant sensors can be achieved if at least two independent measurements can be made. If the difference between the readings of two sensors is above a threshold D_{TH}, a fault is detected although there is no enough information to isolate the fault. When more than two sensors are available, the faulty sensor can be isolated from within the group by detecting which has the biggest difference.[24]

Because the robots are not always close together, the Cooperative FDI layer works only when two or more robots meet, testing the different redundant sensors available. If a fault is detected and only two robots are present, it is assumed that both robots have faulty sensors until a new robot is found.

To identify between additive and stuck type faults, the magnitude of both measurements is stored by each robot. If the difference is similar in two independent tests, it is assumed to be an additive fault, with that difference being the amount of the fault, therefore achieving fault identification.

As there is noise in the measurement made by the sensors, the threshold must be optimized to reduce the false alarm probability. The main problem is that the required threshold might be higher than the accepted fault tolerance, but reducing the threshold will result in a useless system as the number of false alarms will become too high. In this case, the use of a multiple measurements test can solve the problem.

As sensors used in these robots present additive white noise, with known standard deviation σ_I, the difference between the measurements of two sensors, m_i and m_j, has the following error probability distribution:

$$m_i - m_j \sim \mathrm{N}\big(0, \sigma_d^2\big), \quad \text{with} \quad \sigma_d^2 = \sigma_i^2 + \sigma_j^2. \tag{14}$$

If n measurements are done, the standard deviation of the average difference is reduced to σ_n, where

$$\sigma_n^2 = \frac{1}{n}\sigma_d^2. \tag{15}$$

Depending on the desired threshold, D_{TH}, the optimal number of measurements needed in order to have a false alarm probability $1 - P_n$ is determined by the following:

$$n = \frac{\sigma_d^2}{\sigma_n^2}, \quad \text{with} \quad \sigma_n = \frac{D_{\mathrm{TH}}}{D_P}, \tag{16}$$

where D_P is obtained from the standard normal distribution, such that

$$\frac{1}{\sqrt{2\pi}} \int_{x=0}^{D_P} e^{-\frac{x^2}{2}}\, dx = \frac{P_n}{2}. \tag{17}$$

If the robot can only do a limited number of measurements, $n_{\max}$, using (16) and (17), the optimal threshold D_{TH} must be increased to keep the same false alarm probability.

$$D_{\mathrm{TH}} = \sigma_d D_P / \sqrt{n_{\max}}. \tag{18}$$

Every time two robots in the group meet, the sonar and magnetic compass are used to determine the distance and direction between each other. The number of measurements needed is calculated previously using (16), and the average measurement is transmitted between the robots to detect the fault. For the sonar, if the difference between both readings is above $D_{\mathrm{TH}s}$, a fault is detected. For the magnetic compass, the difference between both readings must be 180 [°], so if the difference is outside the range $180 \pm D_{\mathrm{TH}c}$[°], a fault is detected.

4.2. Simulation results

With the same methodology used for the first layer, the capabilities of the Cooperative Layer are tested. The FDI layer is simulated 1000 times with randomly chosen faults, and on each simulation it is assumed that the robots are close together whenever a sensor check is made.

For every sensor, the probability P_n is set to 99.99%. As the standard deviation for the error on the compass is $\sigma_c = 0.15$[°] and for the sonar is $\sigma_s = 0.02$ [m], given that $D_{\mathrm{TH}c} = 0.5$[°] and $D_{\mathrm{TH}s} = 0.05$ [m], then $n = 2$ for the compass and $n = 3$ for the sonar to achieve the desired probability. For this layer, only two criteria are used to test the performance: number of false alarms and confusion matrix.

Because P_n is set to 99.99%, no false alarms appear during the simulations of this FDI layer as false alarms should occur only once every 10,000 times.

The confusion matrix obtained is perfectly diagonal as no confusion can be made in the isolation of the faults. Then, all detected faults are always correctly isolated, and the amount of the fault (for additive faults) is determined with 96% of accuracy.

As the detection and isolation is done only when two robots meet, there is no point in measuring the detection and isolation times.

5. Experimental Results

Due to the computational limitations of our robots, both layers were tested offline using data collected by each robot, which were then processed in a computer.

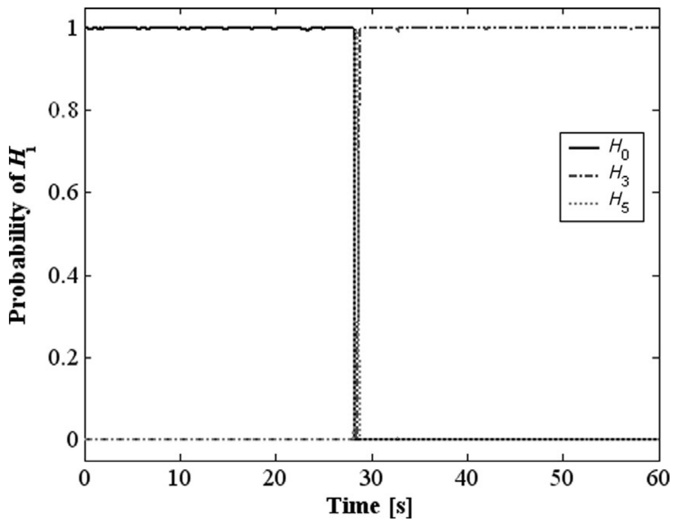

Fig. 4. Experimental result for fault 3: right wheel gets stuck.

The first layer faults are injected in the following way: wheel slippage is done by using the robot over a slippery surface with plastic wheels; the stuck wheel fault is emulated using the robot over thick carpet, this time with rubber wheels; finally, the encoder faults are injected by putting a dark paper between the sensor and the encoder wheel.

Figures 4 and 5 show the results corresponding to two different faults, injected after about 30 [s] of normal operation. After the fault is injected, the probability of H_0 reduces, whereas the probability of the corresponding fault increases. It is important to notice that other hypotheses (H_3 in Fig. 3 and H_4 in Fig. 4) also present a change in their probability. This effect was not observed in the simulations and can be attributed to the slight difference that exists between the parameters of the models used and the ones of the real robot, such as wheel radii and axle length.

It is important to highlight that the detection and isolation capabilities are directly related to the accuracy of the models used. Model-plant mismatches could produce false alarms, increment detection time, or make that some faults remain undetected. These inaccuracies come from several sources such as parameter differences, unknown inputs, and unconsidered nonlinear dynamics, among others, and

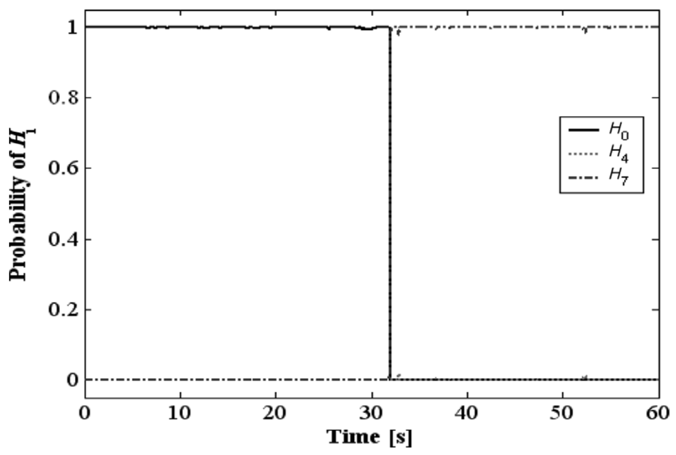

Fig. 5. Experimental result for fault 7: left encoder fault.

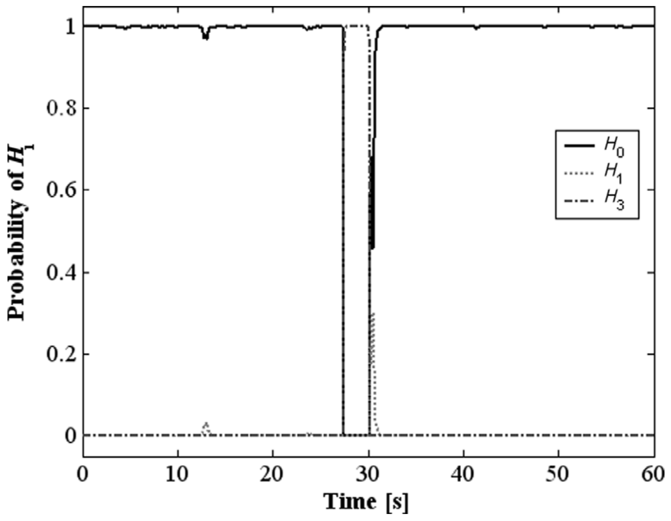

Fig. 6. Experimental result for the "virtual bumper sensor."

sometimes are inherent to the system modeled. In this work, the effect of model-plant mismatches is seen in Figs. 3 and 4, where changes in the probability of hypotheses other than the current operating conditions are seen. However, in this case, the system is able to detect and isolate the fault despite the presence of differences between models. This is due to the detection and isolation criteria used, which require that the probabilities stay above (or below) a threshold for several time steps before raising an alarm or making an isolation decision.

The system was tested 10 times for every fault, correctly detecting and isolating 100% of the cases, which is not surprising given the simulation results, as the amount of times each fault is tested is fairly small.

As this layer is tested offline, the interaction with the control layer must be done artificially. To test the "virtual bumper sensor," the robot is set on a rough surface and is directed toward a wall, making only the right wheel collide with the wall. After 30 [s] of operation the direction of the wheel is inverted, and the robot is positioned at a distance such that after that time it would have already collided with the wall. Figure 6 shows the result of the experiment. After the robot collides the probability of fault 3 increases, but later it decreases when the speed of the wheel is inverted. This indicates that no real fault is present and activates the "virtual bumper sensor" flag.

Before implementing the Cooperative FDI Layer, the sensors were analyzed to make sure that the noise had a normal distribution and to determine its standard deviation, observing that $\sigma_c = 0.157[°]$ and $\sigma_s = 1.31$ [cm].

To test this layer, the robots are set in pairs, detecting each other using the CMOS camera and taking all the measurements needed. Then, an additional robot is used so the fault can be isolated. Every time, the robot randomly chooses one of the possible faults and injects it to the measurement via software. The values are stored in an external memory for later analysis. The number of measurements n_c and n_s are calculated using (16), considering that the tolerated thresholds are $D_{\mathrm{TH}c} = 0.5[°]$ and $D_{\mathrm{TH}s} = 3$ [cm], and that $P_n = 99.99\%$. This sets $n_c = 2$ and $n_s = 3$. The test is done 30 times in total.

The results of the experiments show that no false alarms are activated, and 100% of the faults are correctly isolated. Again it is important to notice that this result is expected as the false alarm probability was set really small and only 30 tests were done. In the case of the additive faults, as the robots stored the amount of the fault injected, it was possible to check the accuracy of the fault identification. In these cases, the amount of the fault is identified with less than 5% error.

6. Conclusions and Future Work

Through this work, a layered architecture for FDI in cooperative mobile robots is successfully designed and implemented. The reliability of the architecture is measured through simulations, showing excellent results as it is capable of detecting all the simulated faults, isolating correctly 99.6% of them. The architecture is then validated by an off-line implementation, which shows similar results, with no false alarms and 100% accuracy in the isolation of the faults.

Furthermore, it is clear that the algorithm used in the Cooperative FDI Layer can be used to isolate faults on GPS and other types of sensors without adding too much complexity to the system, which appears to be quite difficult using other techniques, but having the drawback that at least two robots are needed for detection and three or more for fault isolation.

As future work, an online implementation of the architecture must be made, which means that a more powerful processor must be added to the robots, together with a wireless link that allows information exchange between them. Although these first experiments show good results, more experiments in different environments are required to analyze the robustness of the architecture.

Regarding the architecture itself, more layers can be designed and added to take advantage of the information available at other levels in the control structure and the redundancy existing at the navigation and objective control levels. This could mean for example, monitoring systems that check if the objectives are being achieved, which could help to detect, isolate and even identify new faults, especially those related to robot coordination and trajectory designs.

References

1. J. Carlson, R. R. Murphy and A. Nelson, "Follow-up Analysis of Mobile Robot Failures," *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans (2004) pp. 4987–4994.
2. R. Isermann and P. Balle, "Trends in the Application of Model Based Fault Detection and Diagnosis of Technical Processes," *Proceedings of the IFAC 13th Triennial World Congress*, San Francisco, California (1996) vol. 7, pp. 1–12.
3. M. Basseville and I. V. Nikiforov, *Detection of Abrupt Changes—Theory and Application*, 2nd ed., available at http://www.irisa.fr/sisthem/kniga/ (Nov. 1998).
4. J. Gertler, "Survey of model-based failure detection and isolation in complex plants," *IEEE Control Syst. Mag.* **8**(6), 3–11 (1988).
5. A. S. Willsky, "A survey of design methods for failure detection in dynamic systems," *Automatica* **12**, 601–611 (1976).

6. J. Aravena and F. Chowdhury, "Fault Detection of Flight Critical Systems," *Proceedings of the 20th Conference on Digital Avionics Systems*, Daytona Beach (2001) pp. 1–10.
7. M. Napolitano, D. Windon, J. Casanova, M. Innocenti and G. Silvestri, "Kalman filters and neural-network schemes for sensor validation in flight control systems," *IEEE Trans. Control Syst. Technol.* **6**(5), 596–611 (1998).
8. G. Qin, A. Ge and H. Li, "On-board fault diagnosis of automated manual transmission control system," *IEEE Trans. Control Syst. Technol.* **12**(4), 564–568 (2004).
9. D. Brambilla, L. M. Capisani, A. Ferrara and P. Pisu, "Fault detection for robot manipulators via second-order sliding modes," *IEEE Trans. Indust. Electron.* **55**(11), 3954–3963 (2008).
10. P. S. Maybeck and P. D. Hanlon, "Multiple-model adaptive estimation using a residual correlation Kalman filter bank," *IEEE Trans. Aerosp. Electron. Syst.* **36**(2), 393–406 (2000).
11. R. Isermann, "Model Based Fault Detection and Diagnosis Methods," *Proceedings of the American Control Conference*, Seattle (1995), pp. 1605–1609.
12. J. M. F. Calado, J. Korbicz, K. Patan, R. Patton and J. Sa Da Costa, "Soft computing approaches to fault diagnosis for dynamic systems," *Eur. J. Control* **7**, 248–286 (2001).
13. D. Simon, *Optimal State Estimation: Kalman,* H_∞ *and Nonlinear Approaches* (John Wiley & Sons, Hoboken, New Jersey, 2006).
14. S. I. Roumeliotis, G. S. Sukhatme and G. A. Bekey, "Fault Detection and Identification in a Mobile Robot using Multiple-Model Estimation," *Proceedings of the IEEE International Conference on Robotics and Automation*, Leuven, Belgium (1998) pp. 2223–2228.
15. S. I. Roumeliotis, G. S. Sukhatme and G. A. Bekey, "Sensor Fault Detection and Identification in a Mobile Robot," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Victoria, Canada (1998), pp. 1383–1388.
16. P. Goel, G. Dedeoglu, S. I. Roumeliotis and G. S. Sukhatme, "Fault Detection and Identification in a Mobile Robot Using Multiple Model Estimation and Neural Network," *Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, California (2000), pp. 2302–2309.
17. R. Washington, "On-board Real-Time State and Fault Identification for Rovers,"*Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, California (2000), pp. 1175–1181.
18. W. E. Dixon, I. D. Walker and D. M. Dawson, "Fault Detection for Wheeled Mobile Robots with Parametric Uncertainty," *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Como, Italia (2001) pp. 1245–1250.
19. J. M. Yang, "Fault-tolerant crab gaits and turning gaits for a hexapod robot," *Robotica* **24**(2), 269–270 (2006).
20. J. G. Daniël Karssen and M. Wisse, "Fall detection in walking robots by multi-way principal component analysis," *Robotica* **27**(2), 249–257 (2009).
21. D. G. Michaelson and J. Jiang, "Modeling of redundancy in multiple mobile robots," *Proceedings of the American Control Conference*, Chicago (2000), pp. 28–30.
22. L. Parker, "ALLIANCE: An architecture for fault tolerant multirobot cooperation," *IEEE Trans. Robot. Autom.* **14**(2), 220–240 (1998).
23. R. Tinós, L. E. Navarro-Serment and C. J. J. Paredis, "Fault Tolerant Localization for Teams of Distributed Robots," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Maui (2001), pp. 1061–1066.
24. R. Tinós, M. H. Terra and M. Bergerman, "Fault Tolerance in Cooperative Manipulators," *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington, DC (2002), pp. 470–475.
25. G. Heredia, F. Caballero, I. Maza, L. Merino, A. Viguria and A. Ollero, "Multi-UAV Cooperative Fault Detection Employing Vision Based Relative Position Estimation," *Proceedings of the 17th IFAC World Congress*, Seoul (2008), pp. 12093–12098.
26. G. Heredia, F. Caballero, I. Maza, L. Merino, A. Viguria and A. Ollero, "Multi-unmanned aerial vehicle (UAV) cooperative fault detection employing differential global positioning (DGPS), inertial and vision sensors," *Sensors* **9**, 7566–7579 (2009).
27. M. L. Visinsky, I. D. Walker and J. R. Cavallaro, "Layered Dynamic Fault Detection and Tolerance for Robots," *Proceedings of the IEEE International Conference on Robotics and Automation*, Atlanta (1993), vol. 2, pp. 180–187.
28. M. L. Visinsky, J. R. Cavallaro and I. D. Walker, "A dynamic fault tolerance framework for remote robots," *IEEE Trans. Robot. Autom.* **11**(4), 477–490 (1995).